Amendments to the Specification:

Please replace the paragraph at page 2, lines 2-9, with the following amended paragraph:

In a first aspect of the present invention, a message-passing system comprises a first client system configured to transmit a message packet containing a priority to a second client system, and a second client system configured to receive the message packet from the first client system and process the message packet based on the priority. The message packet can be transmitted from the first client system to the second client system according to a transport protocol. Preferably, the transport protocol is TCP/IP, but it can be ~~NetBUI~~ NetBEUI or any other transport protocol. The message packet can be formatted according to an SGML standard, such as XML. The message packet can comprise text data, a virtual object, or any other type of data.

Please replace the paragraph at page 3, lines 4-11, with the following amended paragraph:

In a second aspect of the present invention, a method of passing a message packet between a first client system and a second client system comprises generating a message packet containing a priority on the first client system, transmitting the message packet from the first client system to the second client system, receiving the message packet on the second client system, and processing the message packet on the second client system according to the priority. The message packet can be transmitted from the first client system to the second client system according to a transport protocol. The transport protocol can be TCP/IP, ~~NetBUI~~ NetBEUI, or any other transport protocol.

Please replace the two paragraphs at page 3, line 27, to page 4, line 14, with the following amended paragraphs:

In a third aspect of the present invention, a sending client system is configured to transmit a message packet containing a priority to a receiving client system, which is configured to process the message packet based on the priority. In accordance with one embodiment, the sending client system comprises a messaging module. The messaging module is configured to assign a priority to a message to form the message packet. The messaging module is further configured to transmit the message packet to the receiving client system according to a transport protocol. Preferably, the transport protocol is TCP/IP, but it can be ~~NetBUI~~ NetBEUI or any other transport protocol.

In a fourth aspect of the present invention, a receiving client system is configured to receive a message packet containing a priority from a sending client system. The receiving client system is configured to process the message packet based on the priority. In one embodiment, the receiving client system comprises a messaging module. The messaging module is configured to receive the message packet from a sending client system according to a transport protocol. The messaging module is further configured to process the message packet based on the priority. Preferably, the transport protocol is TCP/IP, but it can be ~~NetBUI~~ NetBEUI or any other transport protocol.

Please replace the paragraph at page 5, line 17, to page 6, line 3, with the following amended paragraph:

Message-passing systems in accordance with the present invention are particularly useful in manufacturing and other environments. For example, a message-passing system having one or more pieces of manufacturing equipment can be used to monitor the manufacturing equipment and send error, diagnostics, and other messages to a central controller or to a machine monitored by a human operator. Each piece of manufacturing equipment can be coupled to its own sending client system, which monitors a parameter associated with a manufacturing equipment, such as a temperature or pressure. A sending client system can then ~~format~~ generate a message relating to the temperature or pressure, assign the message a priority, format the message, the priority, and other information into a message packet, and transfer the message packet to a receiving client system, such as a control system. The control system uses the priority to process message packets from the sending client system and other sending client systems, generally processing message packets with higher priorities before message packets with lower priorities. The message-passing system can also store log data relating to message packets for archiving and later diagnosis of the functioning of manufacturing equipment.

Please replace the two paragraphs at page 7, lines 10-20, with the following amended two paragraphs:

Hereinafter, a client system generating and transmitting a message packet is referred to as a sending client system. A client system receiving a message packet is referred to as a receiving client system. A message packet generated on a sending client system is referred to as an outgoing message packet. A message packet received on a receiving client system is referred to as an incoming message packet. It will be appreciated that a client system can be configured to both send and receive message packets and thus can be both a sending client system and a receiving client system.

Preferably, a message packet is transmitted from the messaging module 115 to the message server 150 using TCP/IP. It will be appreciated, however, that a message packet can be transmitted from the messaging module 115 to the message server 150 using other transport protocols, including, but not limited to, ~~NetBUI~~ NetBEUI.

Please replace the two paragraphs at page 10, line 15, to page 11, line 11, with the following two amended paragraphs:

Figure 2 is a flow chart 200 for the steps performed by the messaging module 115 of the sending client system 110. First, in the step 210, the messaging module 115 waits for an outgoing message packet that is to be transmitted to the receiving client system 120. As described above, the message packet contains a message that is to be transmitted to the receiving client system 120. The message packet also contains a priority value assigned to the message. The outgoing message packet can be sent from a process running on the sending client system 110, such as a program that reads the temperature on manufacturing equipment coupled to the sending client system 110. The process can be configured to transmit a message packet to the messaging module 115 for transmission to the receiving client system 120 when the temperature exceeds a predetermined level. High priorities can correspond to higher temperatures and thus determine the order in which the message is processed on the receiving client system 120.

Next, in the step 215, the messaging module 115 receives the message and stores the message in its memory. Alternatively, the messaging module 115 receives a reference to the memory shared between a process generating the message and the messaging module 115, where the message is stored. Preferably, the messaging module 115 stores the reference in an outgoing message queue. Next, in the step 220, the messaging module 115 formats the message to generate a message packet containing the message. Preferably, the message packet contains a header and a body containing the message. The header can contain, for example, (1) a priority for the message, (2) a FROM field containing a hostname of the sending client system 110, and (3) a TO field containing a hostname of the receiving client system 120. The sending client system 110 can set the priority to indicate the urgency of the message. For example, a message that manufacturing equipment coupled to the sending client system 110 has an reached an unacceptable temperature will have a higher priority than a message that the manufacturing equipment has not been serviced in one month.

Please replace the paragraph at page 12, lines 24, to page 13, line 5, with the following amended paragraph:

Figure 4 is a flow chart 400 showing the steps executed by a thread running on the message server 150 (e.g., step 320, Figure 3) and used to process a message packet. Referring to Figures 1 and 4, first, in the step 410, the thread reads the message packet. Next, in the step 415, the thread transmits log data to the log server 160. The message server 150 and the log server 160 can be separate machines or the same machine. If the message server 150 and the log server 160 are separate machines, they can communicate using TCP/IP, ~~NetBUI~~ NetBEUI, or any other transport protocol. If the message server 150 and the log server 160 are the same machine, the message server 150 is implemented as a process or set of processes running on the machine, and the log server 160 is implemented as a process or a set of processes running on the machine.

Please replace the paragraph at page 13, lines 12-20, with the following amended paragraph:

Similarly, if the message server 150 and the diagnostics server 170 are the same machine, the message server 150 is implemented as a process or a set of processes running on the machine, and the diagnostics server 170 is implemented as a process or a set of processes running on the machine. The diagnostics server 170 that resides on the same machine as the message server 150 receives incoming message packets containing diagnostic (log) data directly from the message server 150. This exchange can be made using the aforementioned client/server communication methods, which include, but are not limited to, TCP/IP. Alternatively, the diagnostics server 170 and the message server 150 can communicate using shared memory, pipes, or any other method of exchanging data between units of execution running on the same machine.

Please replace the paragraph at page 15, line 12, to page 16, line 5, with the following amended paragraph:

In accordance with the present invention, message packets can have many formats. Preferably, a message packet is an XML document, such as the XML document 600 illustrated in Figure 6 in view of Figure 1. The XML document 600 contains an XML declaration (prolog) 601 and a root element, the message packet element 610 having a tag "messagepkt." The XML declaration 601 <?xml version="1.0"?> indicates that version 1.0 of XML is being used. The message packet element 610 contains a header element 615 with the tag "header" and a body element 620 with the tag "body". The header element 615 contains (1) a priority element with the data "5 ", indicating that the attached message (discussed below) has a priority of 5; (2) a ToIP element with the data "process_monitor", indicating that the attached message is to be sent to a receiving client system having the hostname "process_monitor"; (3) a ToPORT element with the data "1880", indicating that the attached message is to be sent to the port 1880 on the receiving client system 120; (4) a FmIP element with the data "machine1," indicating that the sending client system [[120]] 110 has the hostname "machine1"; and (5) a FmPORT element with the data "1881", indicating that the sending client system [[120]] 110 will send the message from its port 1881. It will be appreciated that the above elements and their associated data are used for illustration only and are optional or may have different values. It will also be appreciated that a message server can be used as a name server to translate hostnames to IP addresses in dotted decimal notation, if needed. Alternatively, if a message server is not used, the data in the ToIP element and FmIP element can be any identifiers that a name server, such as a domain name server used on the Internet, can use to forward a message packet from a sending client system to a receiving client system.

Please replace the paragraph at page 20, lines 13-23, with the following amended paragraph:

Figure 9 shows a load-balanced message-passing system 900 that balances message packet transmissions between two message servers 950 and 955. Thus, when one message server is busy processing a message packet, a message packet can be sent to and processed by another message server, thus increasing overall system throughput. The load-balanced message-passing system 900 comprises a first client system 910 having a messaging module 915 and a second client system 920 having a messaging module [[915]] 925; a load-balancer 940 coupled to both the first messaging module 915 and the second messaging module 925; [[a]] the first message server 950 and [[a]] the second message server 955, both coupled to the load balancer 940; a log server 960 coupled to both the first message server 950 and the second message server 955 and having a messaging module 965; and a diagnostics server 970 coupled to both the first message server 950 and the second message server 955 and having a messaging module 975.